L2: Entry 1 of 1 File: USPT Jul 6, 1999

DOCUMENT-IDENTIFIER: US 5920873 A
TITLE: Data management control system for file and database

Detailed Description Text (113):
The information for each process can be entered/edited individually on a menu containing all the above fields or a utility exists to load "process groups" which are pre-defined library controlled processes. The Data Manager simply selects a process group and attaches it to the appropriate data type, level and version. The process groups are ASC based files which contain the necessary process information in a prescribed format. They can be created using any ASC editor.

Detailed Description Text (147):
Create and monitor a Bill of Materials. The utility offers two modes of operation. In the first, the user identifies the Bill of Materials, and enters the names of all design components to be added as members. This same utility will display any existing information for a BOM, so members can be modified or deleted. For each member, the user must indicate whether it's an input, output or support member. For an existing BOM, a function exists to revalidate all members, but this can only be done by the BOM owner. The second mode builds the BOM by reading all the information from an ASC text file written in a prescribed format. This mode can be used by designers, Data Managers, and third party tools. Regardless of how the BOM is created, a newly created BOM will result in the valid flags being set for all members. The user who creates the BOM using the first mode is automatically the owner, whereas the input file used for the second mode contains the owner information.

Detailed Description Text (151):
The DCS provides a mechanism which permits access to all process and pseudo process results through the World Wide Web. Key quality control indicators can be exported out of the DCS into an accessible format by users on the WWW. Usually these results would exist in a secure repository which could only be accessed by WWW users who are working on the project. In addition to accessing information, the ALMs can receive special e-mail requests from users to perform these tasks:

Detailed Description Text (185):
Field 22201 holds the name of the file to be promoted. If a single file is being promoted, the name is typed in directly. If the user desires a selection list, it is automatically invoked by simply leaving the field blank and completing the remainder of the form. Upon hitting Enter, a library search would be employed to obtain a selection list of files. The third method is to promote a group of files via a text-based list, edited in advance, in a prescribed format. This is covered below in the explanation of user options.

Detailed Description Text (443):
Step 29158 is designed to handle requests which Create a Structure File. Our preferred embodiment uses Structure Files to supplement the Structure Tables in the Control Repository. These files contain a formatted list of all the Levels and Versions installed for this Package, their repositories, and the information linking the Level and Version tree. This permits many of the Data Management functions to reference this file instead of querying the repository, thereby increasing availability and possibly improving performance. In order to assure that these files are kept in sync with the Control Repository, any changes made by the Data Manager to the library structure result in a Create Structure File Request being sent to the library's main ALM. Upon receiving it, Step 29159 is invoked to Update the Structure File using the latest information in the DMS. This step extracts the structure information from the Control Repository and writes it into the Structure File with the proper format.

Detailed Description Text (599):
Additional functions are provided on Menu Bar 72926. Upon selecting Filter, the user is presented with a dialog box offering radio buttons to select either Hierarchy or No Hierarchy. These buttons determine whether the model information is displayed in Window 72929 with all Anchors and Components or just the Anchors. There's also a push button to select Indented display which formats the output in a manner such that every hierarchical level of the Model is indented. Additionally, push buttons exist to allow the user to format the display information. These buttons are used to display any combination of our PFVL with the following:

Detailed Description Text (616):
Reports Offers a variety of formatted reports that can be generated for the current Model. Our preferred embodiment includes replication of window 72929 with hierarchical or non-hierarchical formatting. Additional information beyond that shown in window 72929 may also be included in the reports. Some examples are model reference numbers, date/time of invalidation, user causing the invalidation, reason for invalidation, etc. Our Aggregation Manager also contemplates the export of the report data in popular commercial spreadsheet and database formats.

Detailed Description Text (617):
Print Prints the current view in window 72929 to an attached printing device or to a file in a printer-specific format (ie. PostScript).

Detailed Description Text (630):
The second method of inputting data into a Model is via the ASC List selection on cyclic field menu 72924 of FIG. 75. Upon hitting enter, the user is presented with a dialog box requesting the name of a formatted text file like the one shown in FIG. 78. Our preferred embodiment uses the following format, although one skilled in the art would clearly see that the same information can be conveyed to the program in numerous other formats. FIG. 78 indicates five records to show a sample Model containing an Anchor in Record 72951 and four Components in records 72952 thru 72955. Our Aggregation Manager supports any number of records, and the Anchor is not required to be listed first. The preferred embodiment shows each record comprising the same format which consists of six or seven tokens. The first six tokens are required and represent:

Detailed Description Text (640):
Returning to FIG. 75, the user's third method for inputting Model data is via the Data Entry choice on cyclic menu 72924. This results in a simple data entry screen where the user must interactively type in the same information which is provided by the text file method previously described. In our preferred embodiment, the information is entered in the same format as FIG. 78 which enables the same algorithm to process it. If the user has File Reference numbers on hand, they can be entered as optional seventh tokens. Once the user finishes typing all the desired records into the data entry screen, a Commit button is provided to install the modifications into the DMS. Once again, if an Anchor is detected, it must match the Model identified in FIG. 75. Also, if a model already exists, the user is given the opportunity to replace it or append the information to it.

Detailed Description Text (767):
Record 38433 is identical to record 38429 but represents the file information for the second file being processed by this Bucket. Records 38434 thru 38436 are identical to records 38430 thru 38432, except they pertain to the second file. As expected the only difference between these sub-sections is the File Reference token. The Process and Log Reference numbers are identical in records 38430 and 38434 since these records refer to Process.sub.-- 12. The same is true for records 38431 and 38435, and also for records 38432 and 38436. The arrangement of the information employed in our invention permits a single format to convey an endless combination of Library Process configurations and parameters.

Detailed Description Text (785):
At this point, Step 38510 is again executed to establish a Bucket Loop In Step 38543 a File List is created by writing the names of all input files associated with the current Bucket into a special File List. This information is different from the Master File List contained in the Job File in two ways. First, this list only contains files used in the current Bucket which may be a subset of the Master File List. Second, the format is more convenient for the intended user of the list. The sole purpose of this file is to act as the input list to drive the actual

Library Processes in the Bucket.

Detailed Description Text (840):
FIG. 88 illustrates an example of a PED file. This example shows 8 tokenized records. Record 39201 must be the Header Record which consists of 3 tokens. The first is an "H" to indicate it's the header record. The second is the filetype that would be used to create a Bill of Materials in the library, if a one is desired. The third token is used to determine the type of BOM if one will be created. Possible values are FULL, INC (for incremental) or NONE. Records 39202 thru 39204 are the Member Records which always begin with an "M" as the first token. The number of member records depends on the task that is executing. There is one record for every input and output file that will be libraried. The second token of a Member record is the actual filename. In a network environment, this is the full path name since input and output files may be scattered across multiple directories. The third token is the library filetype which may not be obvious from the filename. Since the DMS requires that all data be uniquely identifiable by Package, Filetype, Version, and Level (PFVL), the filetype must be determined during PED creation. The fourth token is the BOM Member Type. Possible values are "A" for Anchor file, "I" for Input file, "O" for Output file and "S" for Support file. If the process requires a BOM to be created, the Member records contain all the necessary information. The fifth token is the File Identification Code (FIC). In our preferred embodiment, a Cyclic Reduncany Check is used as the FIC, but any available algorithm that assigns unique identifiers to files, based on their contents, could be used. Record 39205 of the PED file in FIG. 88 is the Result Record. It contains an "R" as the first token. The second token is the name of Process within the Control Repository where the result should be recorded. The third and fourth tokens contain the filename and library filetype of the file for whom the process result should be associated with. The fifth token is the actual result and must be in a form that the Control Repository can accept. Records 39206 and 39207 are Support Records whose first token is always a "K". These are important files that are used in the task, but won't be transported to the library. The most frequent examples are executables, binaries, technology and rules files. These records are simple in format. Besides the first token, the second token is the name of the file, and the FIC is the third token. The last record, record 39208 in FIG. 88, is the Master Pedigree Record. It's denoted by a "P" as the first token. The second token is the filename of the Pedigree file and the third token is the library filetype of the PED file. The fourth token is a special FIC. Aside from the first record being a Header record, the remaining records can be in any order. In our preferred embodiment it is desirable to have the Master Pedigree Record as the last record to simplify the special FIC creation.

Detailed Description Text (872):
If the CRC check passes, the program asks the question in Step 39422 "is this a New File". A new file is one that is not currently being tracked by the DMS. If the answer is "yes", some additional checks must be performed. Step 39423, Lock Check is designed to ensure that the user has the proper authority to promote the current file into the DMS. Since External Data Control substitutes for a formal promotion, it must comply with certain data integrity rules. The Lock Check essentially ensures that the file is not prohibited from entering the library due to an existing overlay or processing lock. It then checks to see if the file is "owned" by anyone. If so, it must be the user or someone for whom the user is a surrogate. In the latter case, the lock would be reset in favor of the user, and notification would be sent to the original owner. Once the Lock Check is cleared, Step 39424, Inst List is executed. Here the file is added to a list of files to be installed into the library at the end of the processing. The last thing the program must do with an "M" record is Add to BOM in Step 39425. In the preferred embodiment, this is done by parsing the file information from the second and third tokens into the proper format to write it into a temporary file. This file is used in a later by the Aggregation Manager's API which creates BOMs for the DMS. The fourth token, which is the BOM Member Type, is also written along with the file information.

Detailed Description Text (1009):
An ISO approved verification audit requires a ISO Quality Record. This task requires a DMS capable fo storing results from tasks along with the proper pedigree information for the files used to run the tasks. It can then be enhanced to produce output in a comparable format to the ISO Quality Record In this connection we use our process management function and our release manager.

Detailed Description Text (1071):
ISO approved verification audits, which we discuss under Section 6.6, requires a DMS capable of

storing results from tasks along with the proper pedigree information for the files used to run the tasks. It can then be enhanced to produce output in a comparable format to an ISO Quality Record. The current ViewLogic system offers no such mechanism, and our process management function and our release manager methods needs to be employed. While we have described our preferred embodiments of our inventions, it will be understood that those skilled in the art, both now and in the future, may make various improvements and enhancements which fall within the scope of the claims which follow. These claims should be construed to maintain the proper protection for the inventions first disclosed.

Previous Doc Next Doc Go to Doc#